

FIG. 1

```
100 struct S {
101     S(); ~101
102     ~S(); ~102
103 };
104 struct T {
105     T(); ~104
106     ~T(); ~105
107 };
108 void woof();
109 ...
110 L1: {
111     T ant; ~107
112     try {
113         if( x>0 ) {
114             S boa; ~110
115         }
116         } catch( int y ) {
117             S cat; ~112
118             T dog; ~113
119             woof(); ~114
120         }
121     }
122 L2:;
```

```
{ #include <setjmp.h>
    struct EH_item {
        struct EH_item * next; ~201
        enum {DESTROY,TRY} tag; ~202
        union {
            struct {
                void * object; ~203
                void (*dtor)(); ~204
            } destructor;
            struct {
                jmp_buf buffer; ~205
                struct handler_spec* handlers; ~206
            } try_block;
        };
    };
}

struct EH_item * EH_stack_ptr; ~207
```

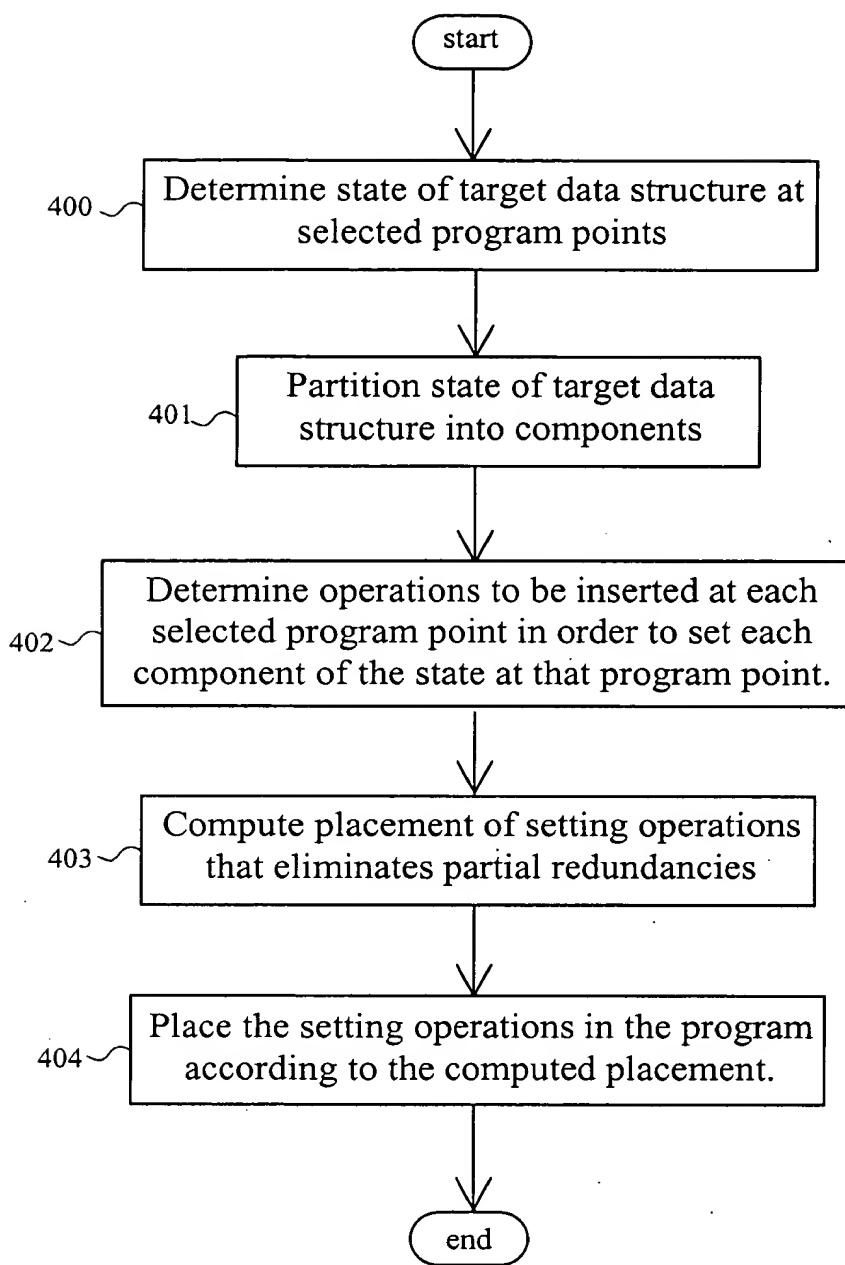
FIG. 2

FIG. 3
PRIOR ART

```

struct EH_item ra, rb, rc, rd, re, rt;
L1:
303 ~T(&ant);
304 ~ra.kind = DESTROY;
    ra.destructor.object = &ant; ra.destructor.dtor = &~T;
306 ~ra.next = EH_stack_ptr; EH_stack_ptr = &ra;
307 ~rt.kind = TRY;
    rt.next = EH_stack_ptr;
    rt.try_block.handlers = ...;
31 ~rt.next = EH_stack_ptr; EH_stack_ptr = &rt;
31 ~if( setjmp( rt.try_block.buffer)==0 ) {
    if( x>0 ) {
        313 ~S(&boa);
        314 ~rb.kind = DESTROY;
            rb.destructor.object = &boa; rb.destructor.dtor = &~S;
            rb.next = EH_stack_ptr; EH_stack_ptr = &rb;
        31 ~EH_stack_ptr = EH_stack_ptr->next;
            ~S(& boa);
    } else {
        S(cat);
        rc.kind = DESTROY;
        rc.destructor.object = &cat; rc.destructor.dtor = &~S;
        rc.next = EH_stack_ptr; EH_stack_ptr = &rc;
        T(&dog);
        rd.kind = DESTROY;
        rd.destructor.object = &dog; rd.destructor.dtor = &~T;
        rd.next = EH_stack_ptr; EH_stack_ptr = &rd;
        woof();
        EH_stack_ptr = EH_stack_ptr->next;
        ~T(&dog);
        EH_stack_ptr = EH_stack_ptr->next;
        ~S(& cat);
    }
} else {
    S(&elk);
    re.kind = DESTROY;
    re.destructor.object = &elk; re.destructor.dtor = address of ~S();
    re.next = EH_stack_ptr; EH_stack_ptr = &re;
    ~S(&elk);
    EH_stack_ptr = EH_stack_ptr->next;
}
342 ~EH_stack_ptr = EH_stack_ptr->next;
343 ~EH_stack_ptr = EH_stack_ptr->next;
34 ~ ~T(ant);
L2:

```

**FIG. 4**

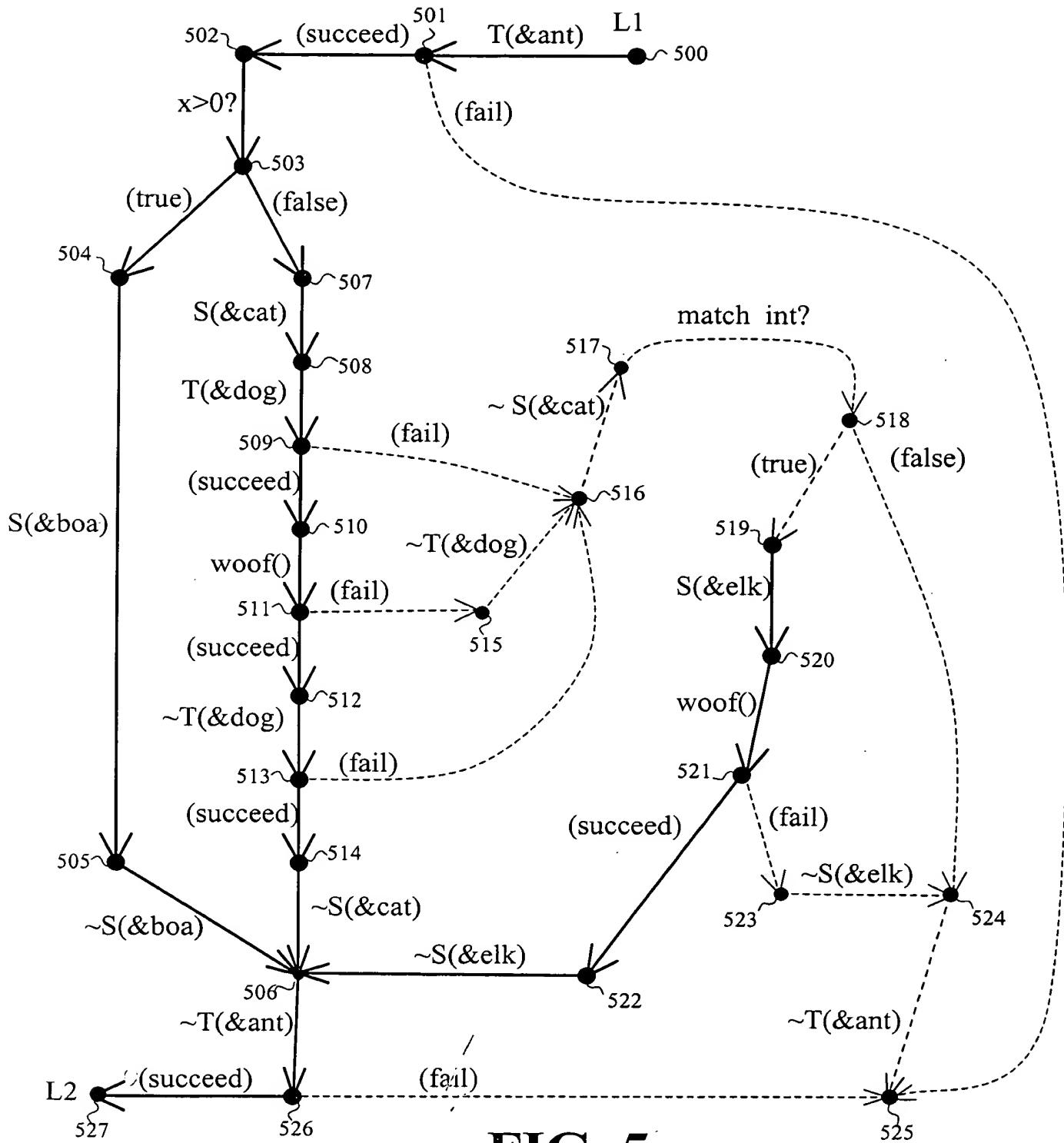
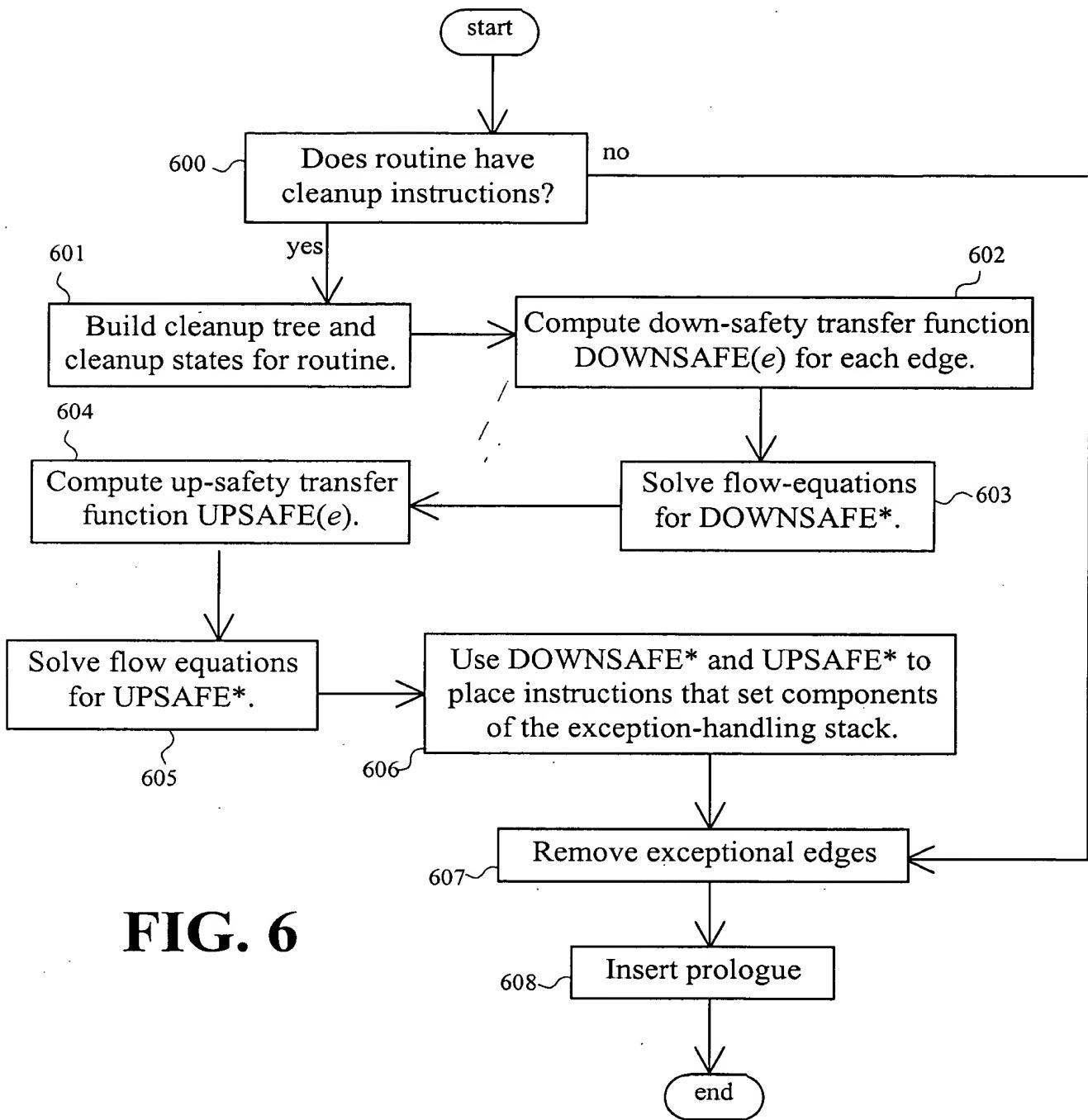


FIG. 5



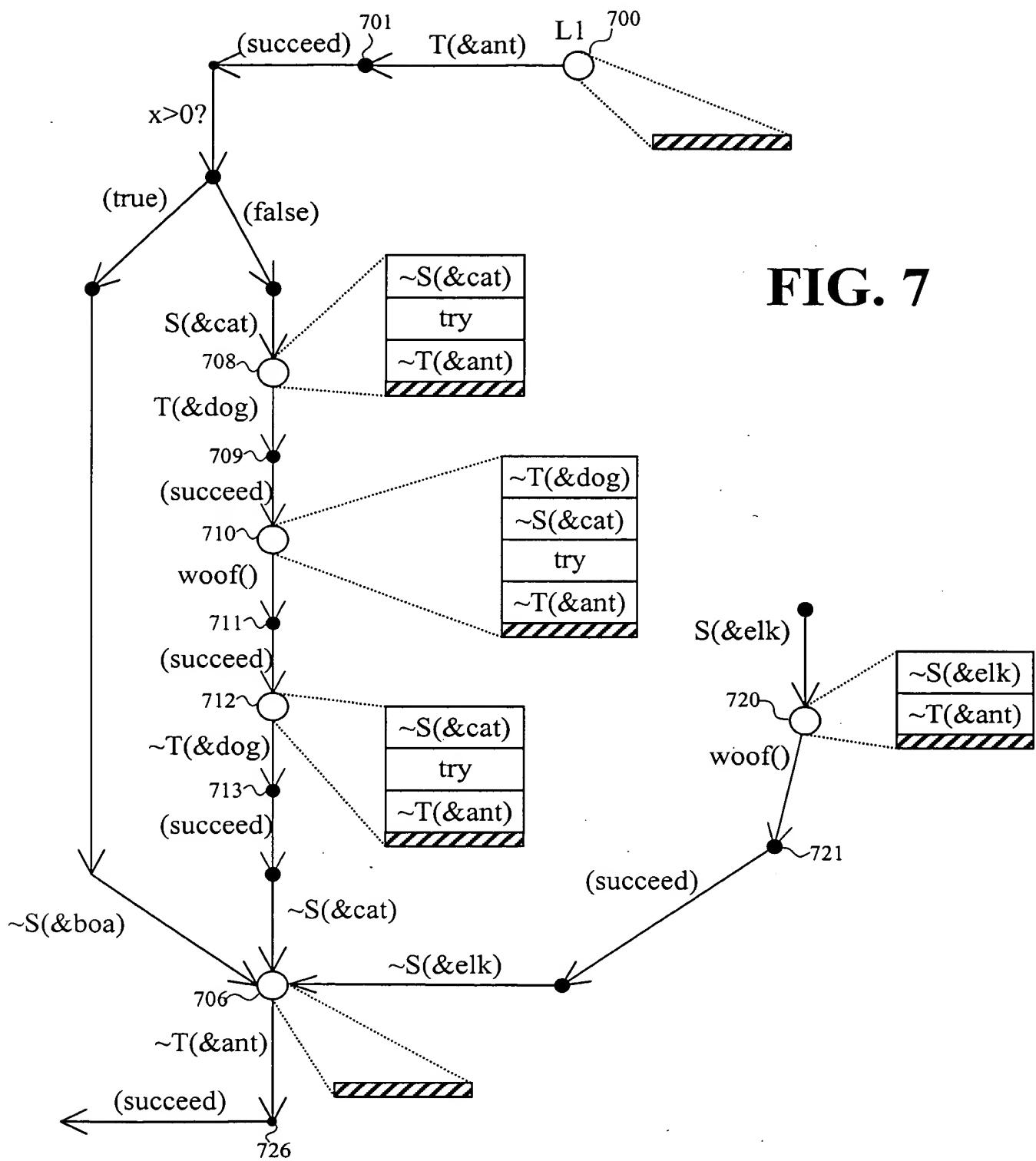
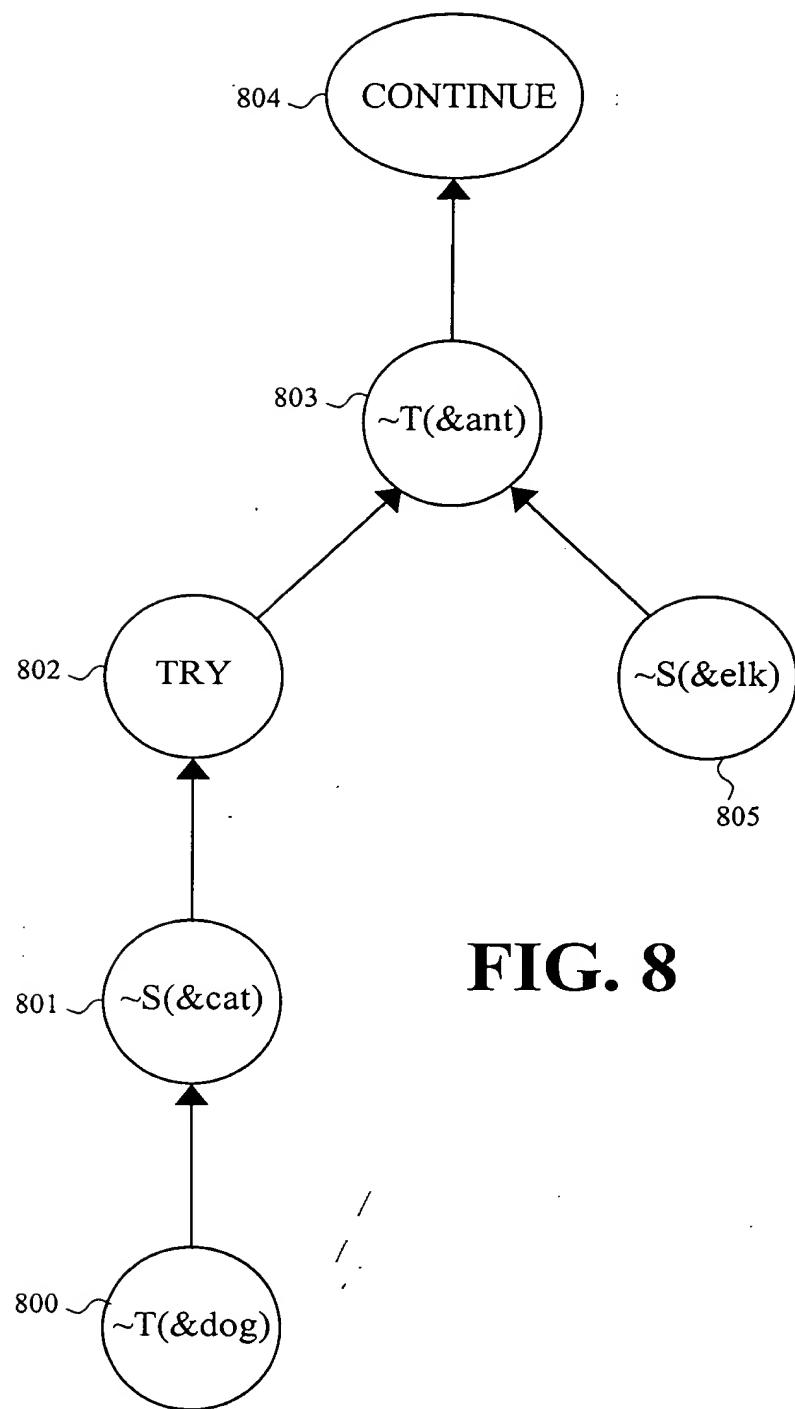


FIG. 7

**FIG. 8**

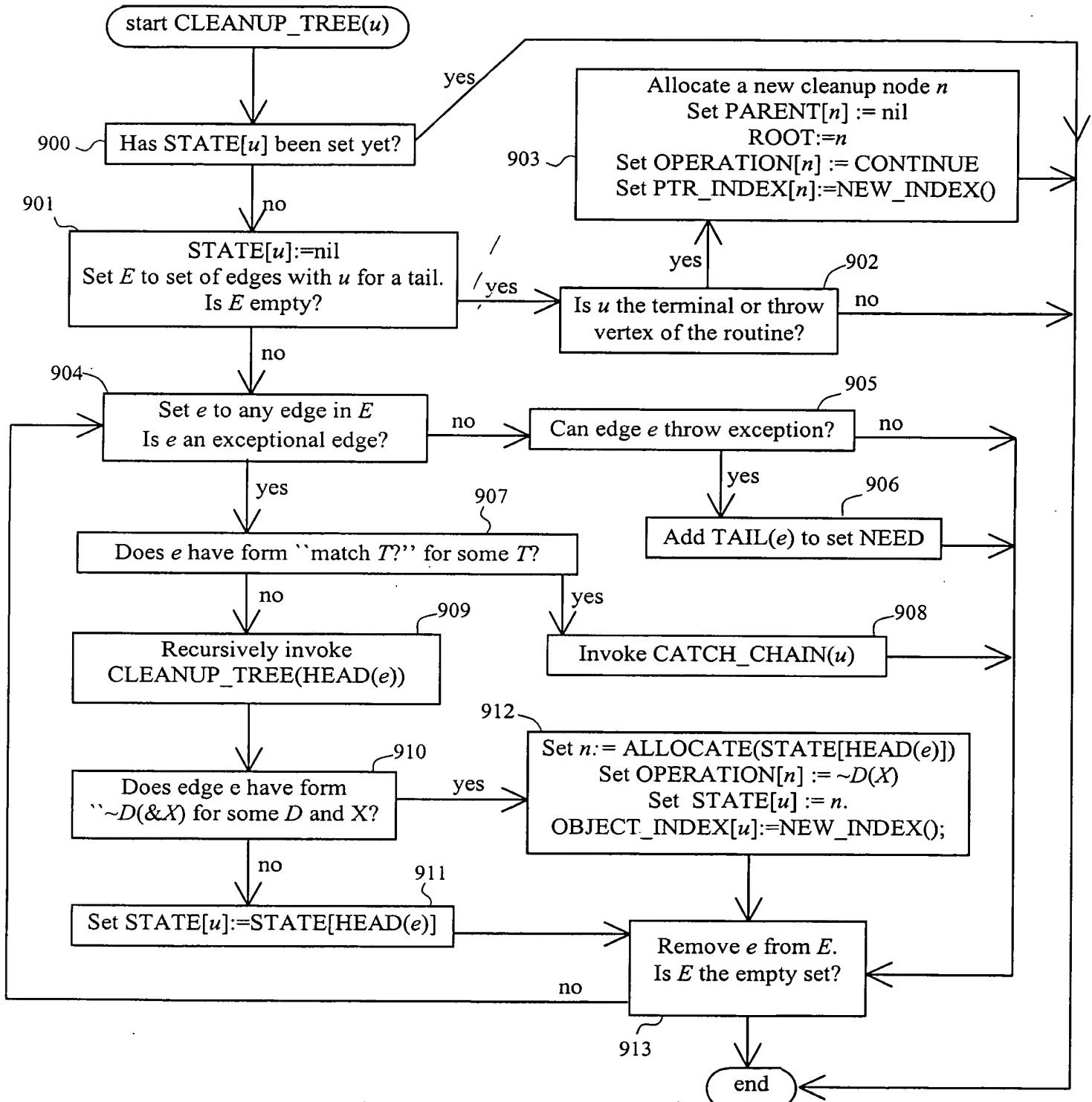


FIG. 9

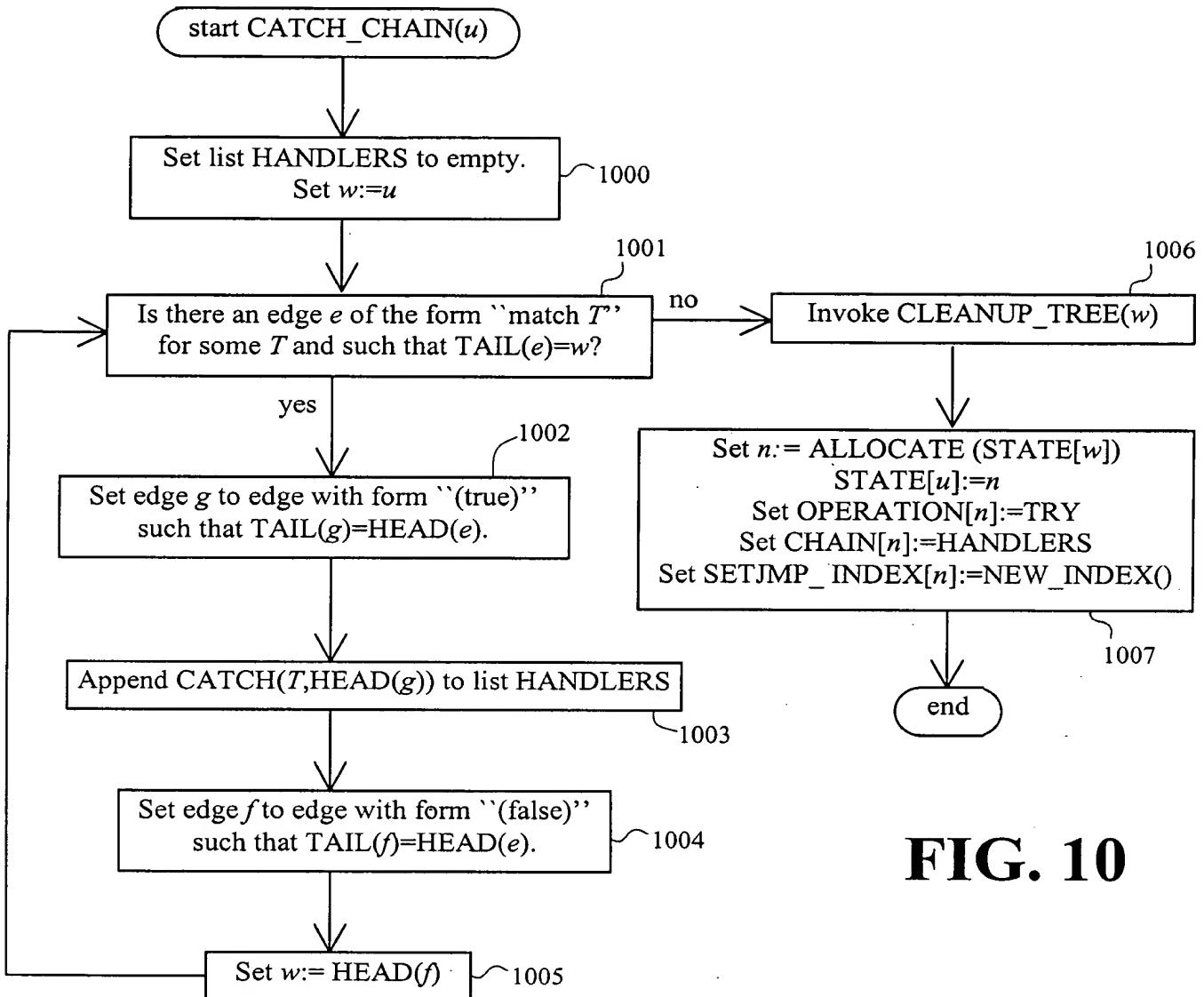
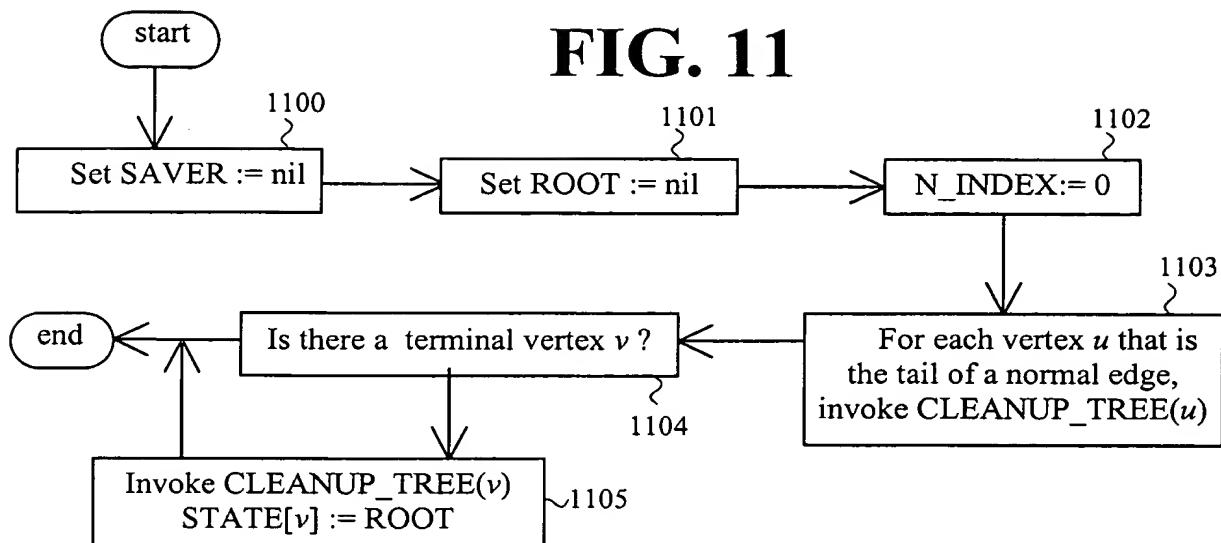
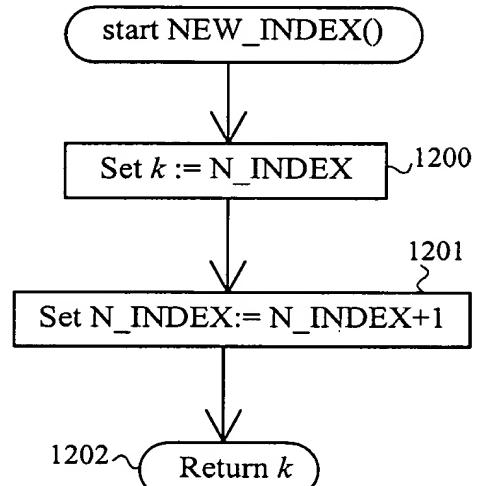
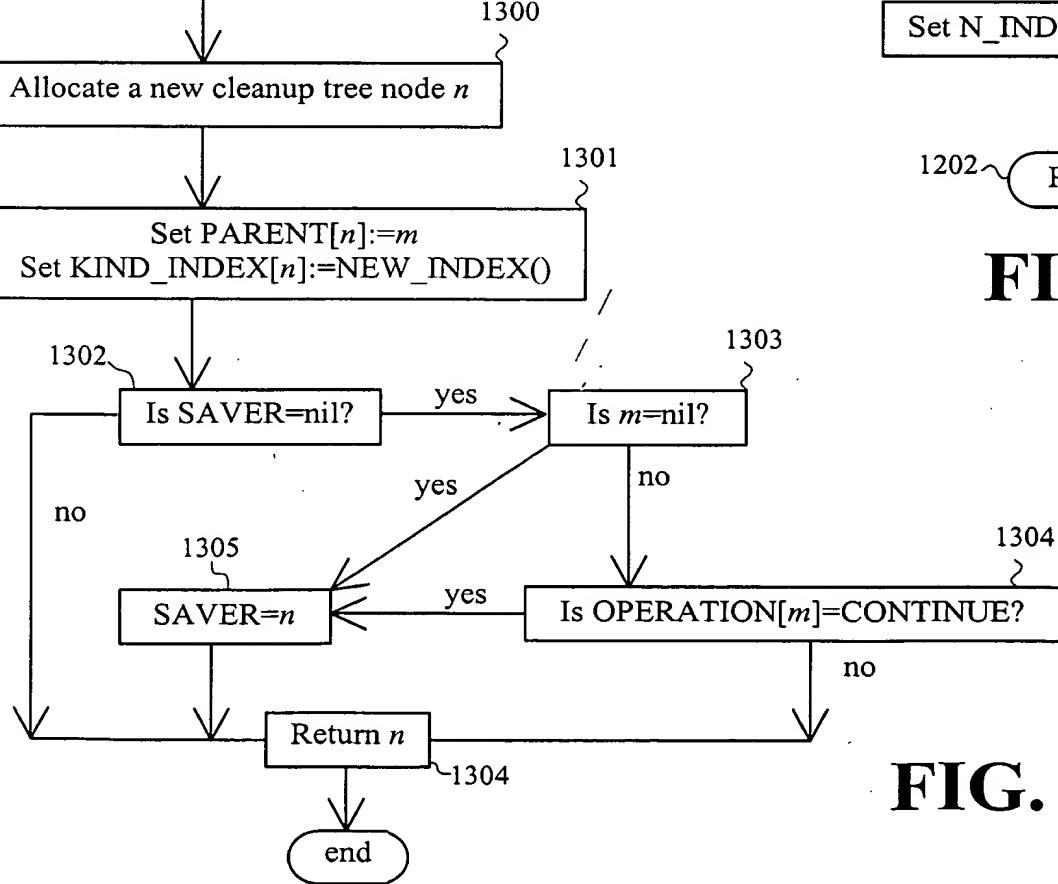


FIG. 10

FIG. 11**start NEW_INDEX()****FIG. 12****FIG. 13**

12/25

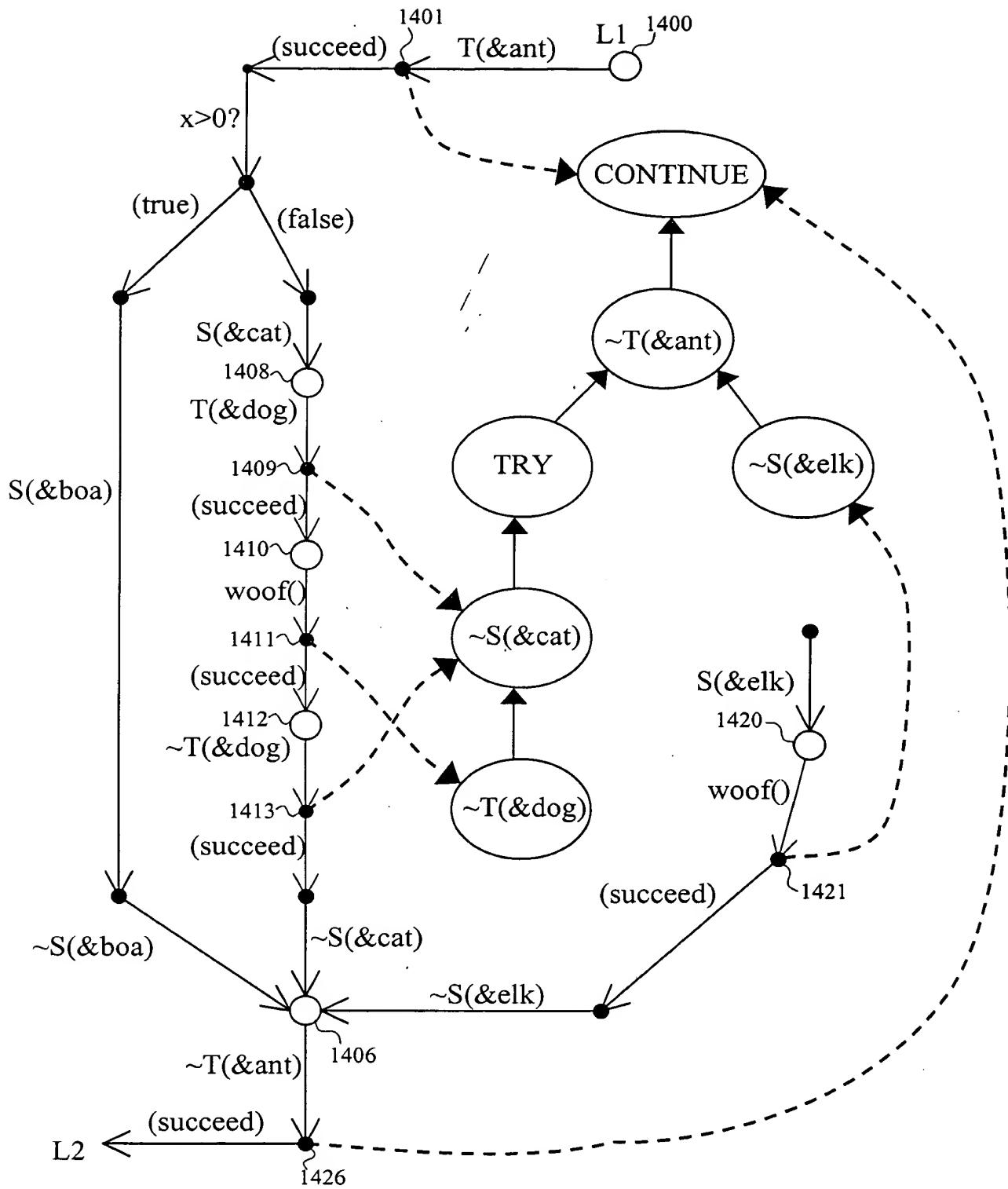
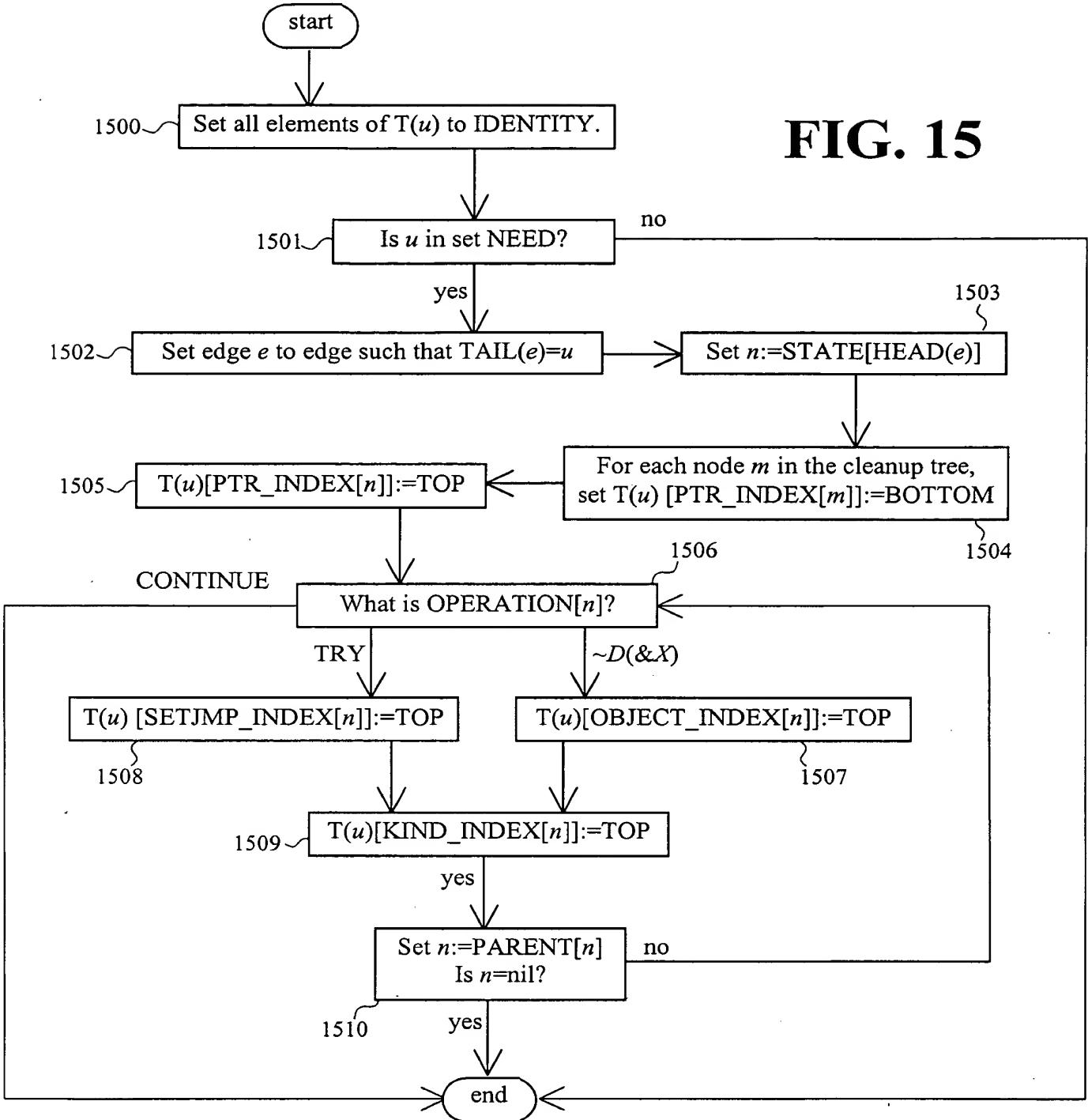


FIG. 14



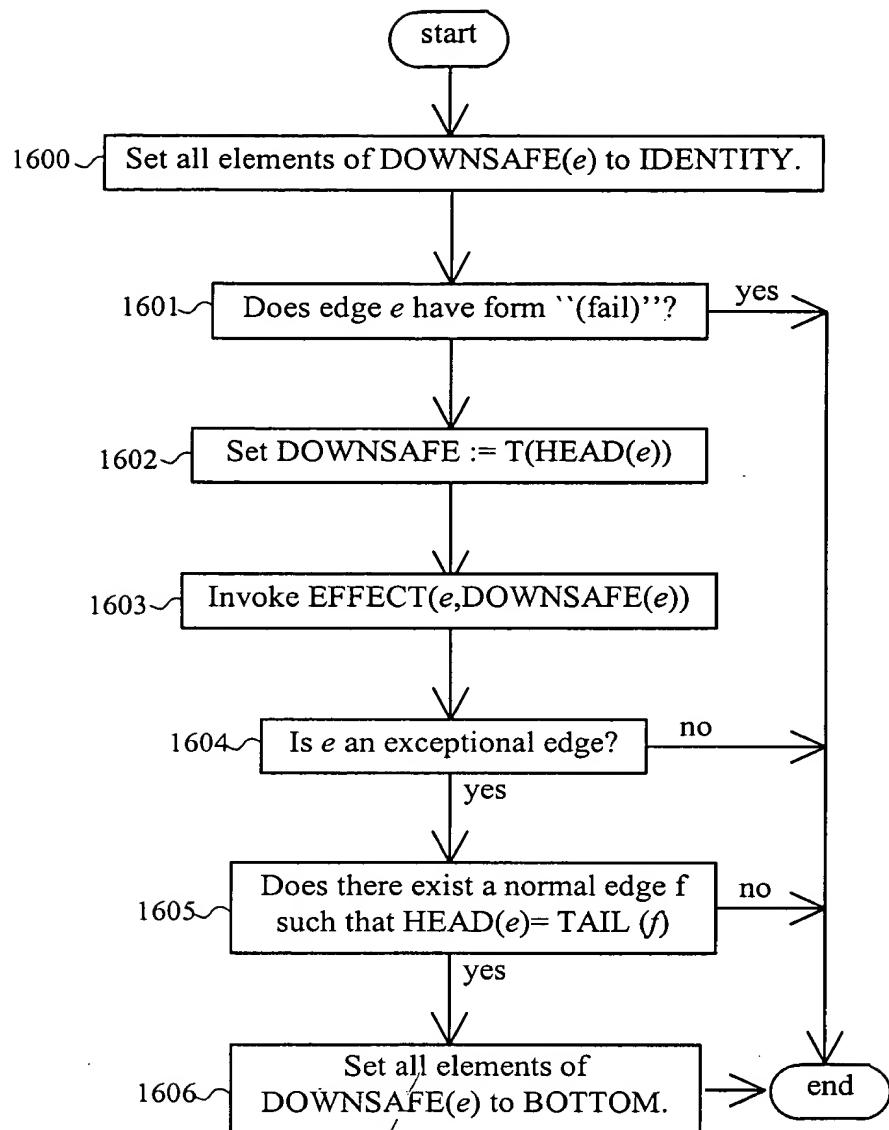


FIG. 16

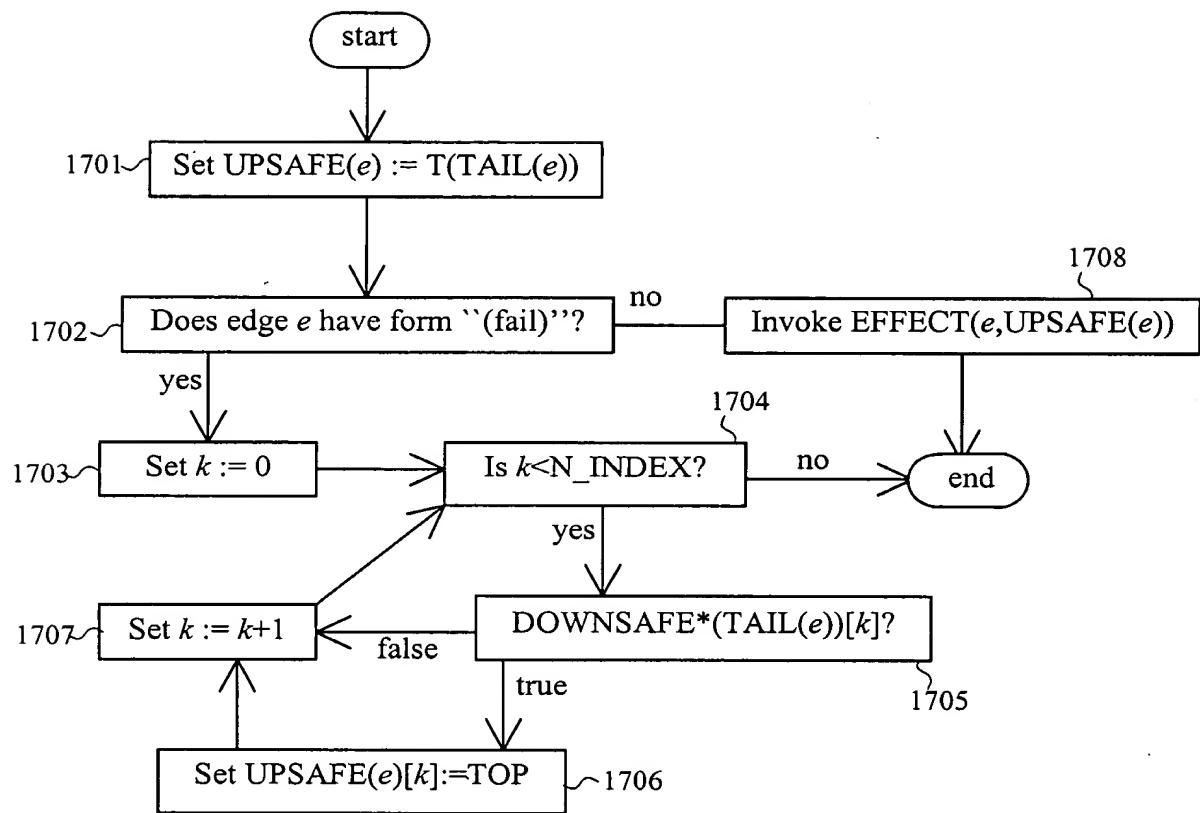


FIG. 17

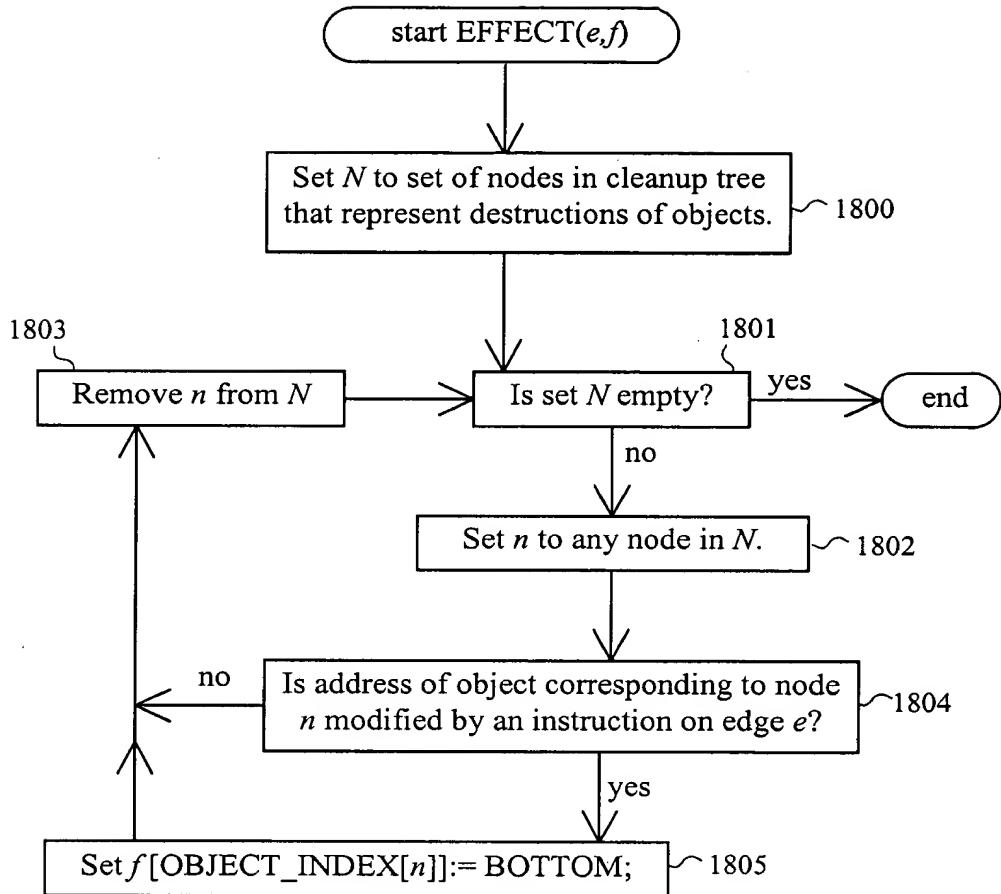


FIG. 18

FIG. 19

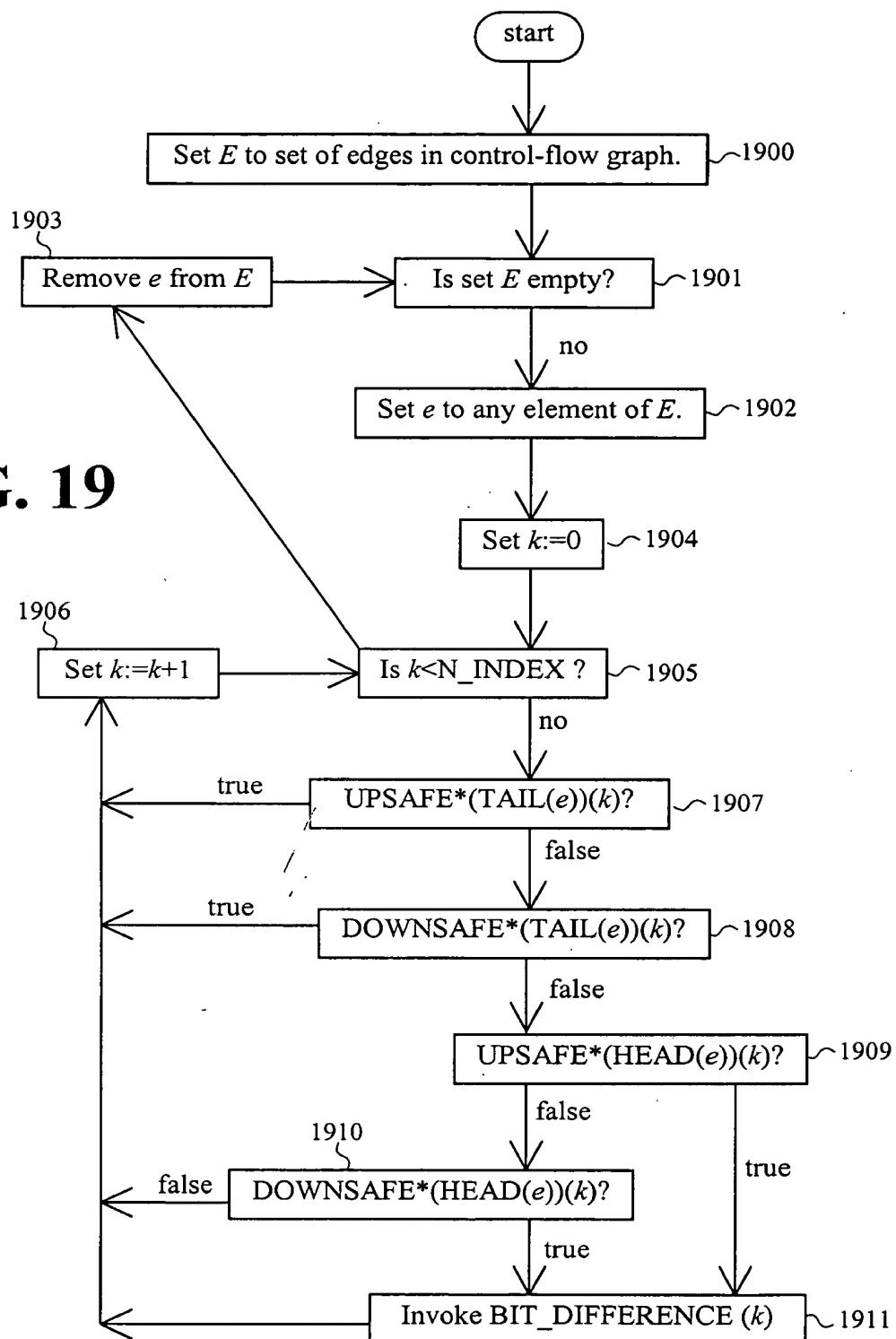


FIG. 20

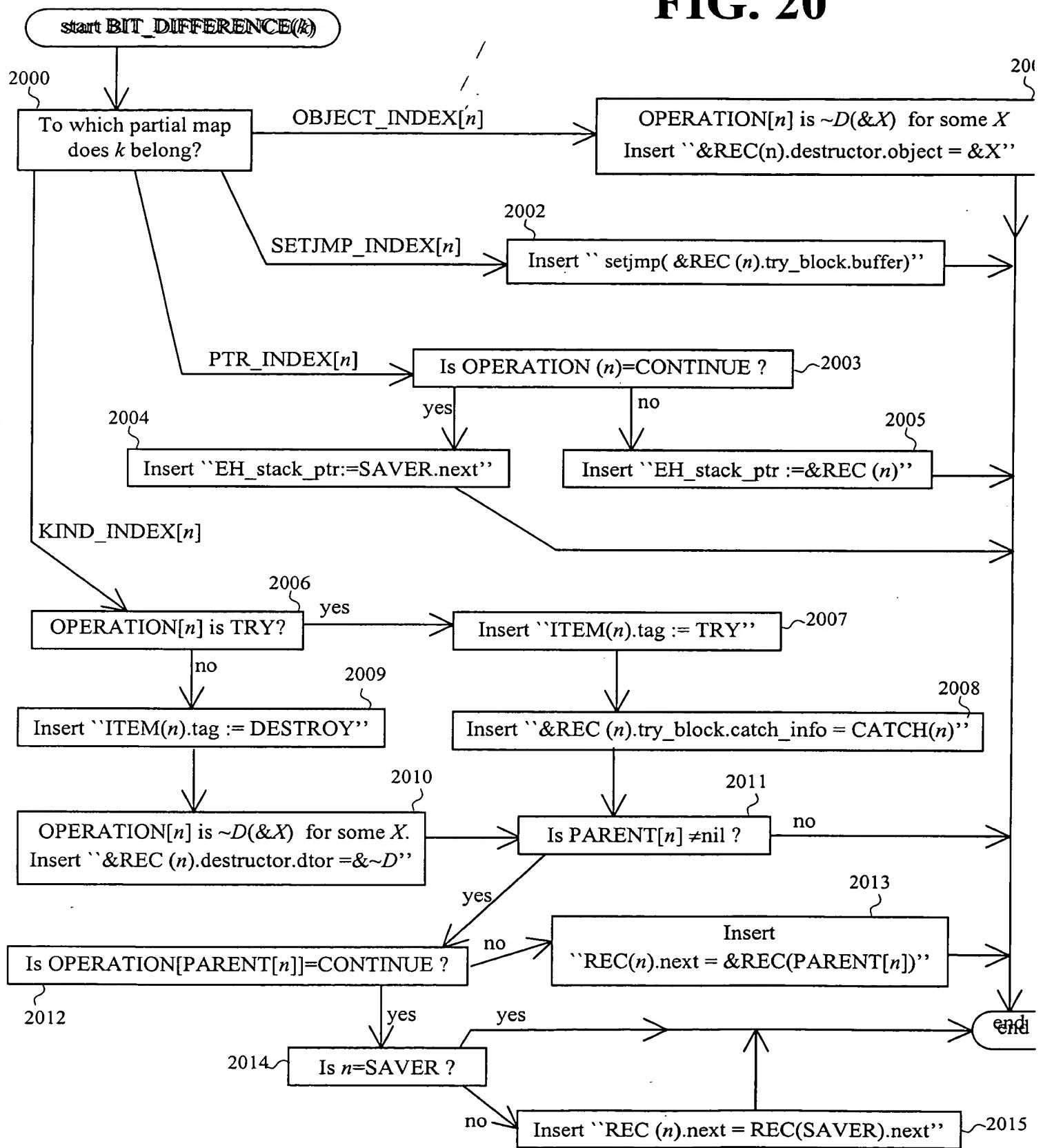


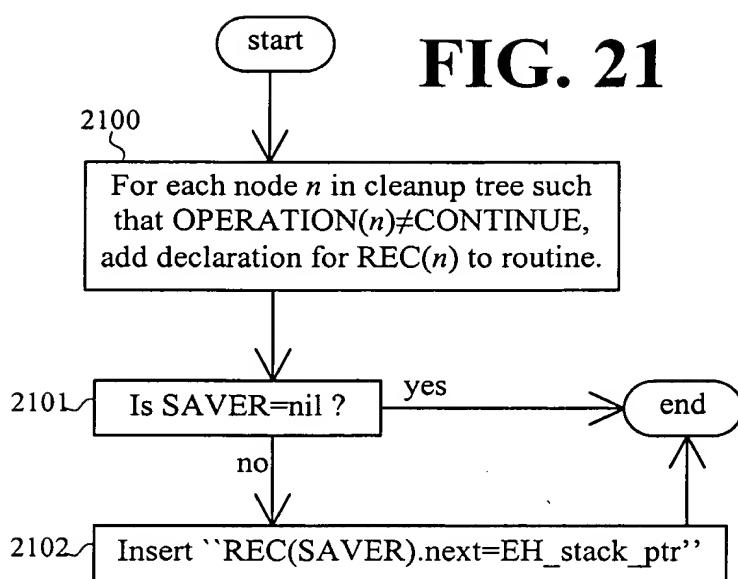
FIG. 21

FIG. 22

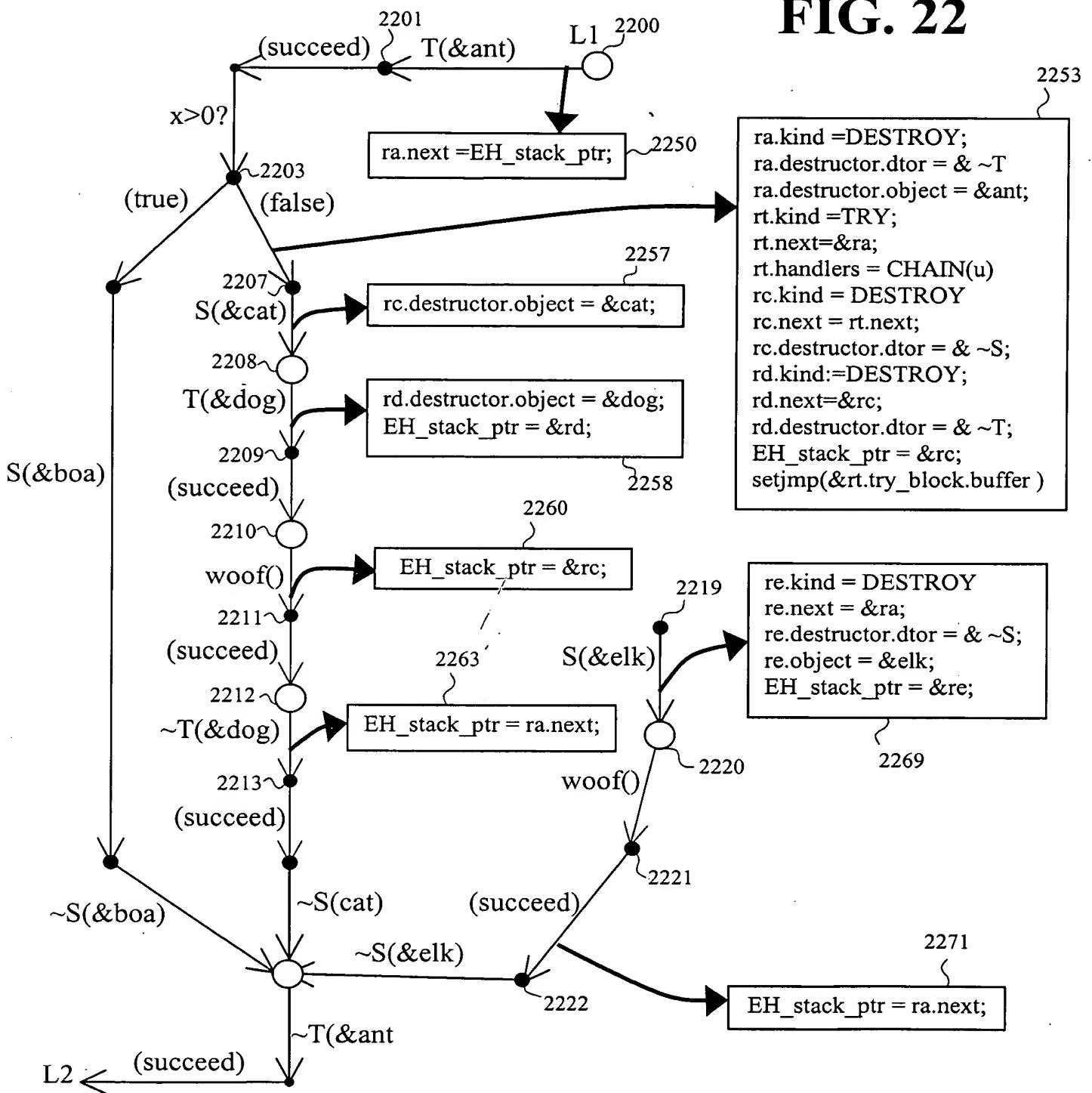


FIG. 23

```

struct EH_item ra, rb, rc, rd, re, rt; ~2301
L1: /
ra.next = EH_stack_ptr; ~2303
T(&ant);
if( x>0 ) {
    S(&boa);
    ~S(&boa);
} else {
    ra.kind = DESTROY;
    ra.destructor.dtor = &~T;
    ra.destructor.object = &ant;
    rt.kind = TRY;
    rt.next = &ra;
    rt.try_block.handlers = ...;
    rt.next = &ra;
    rc.kind = DESTROY;
    rc.destructor.dtor = &~S;
    rc.next = &rt;
    rd.kind = DESTROY;
    rd.destructor.dtor = &~T;
    rd.next = &rc;
    eh_stack_ptr = &rc;
    if( setjmp( &rt.try_block.buffer)==0 ) {
        S(&cat);
        rc.destructor.object = &cat;
        T(&dog)
        rd.destructor.object = &dog;
        eh_stack_ptr = &rd;
        woof();
        EH_stack_ptr = &rc;
        ~T(&dog);
        EH_stack_ptr = ra.next;
        ~S(&cat);
    } else {
        re.kind = DESTROY;
        re.next = &ra;
        re.destructor.dtor = &~S;
        eh_stack_ptr = &re;
        S(&elk);
        re.destructor.object = &elk;
        woof();
        EH_stack_ptr = ra.next
        ~S(&elk);
    }
}
~T&ant)
L2:

```

FIG. 24

```

struct R {
    R(); ~2402
    ~R() throw(); ~2403
};
...
{
    i=0;
    do { ~2408
        R fox; ~2409
        woof(); ~2410
        i=i+1;
    } while( i<100 ), ~2412
}

```

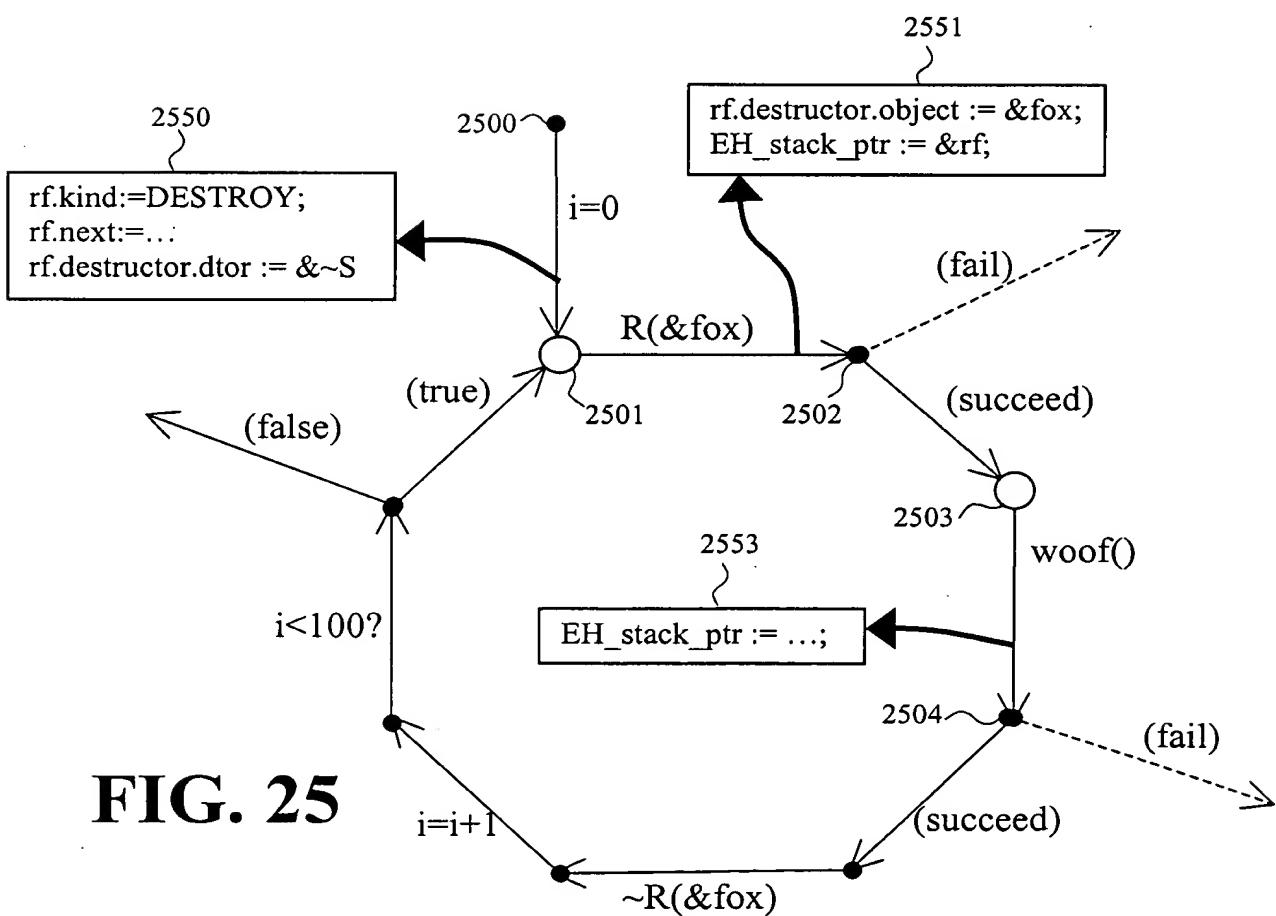
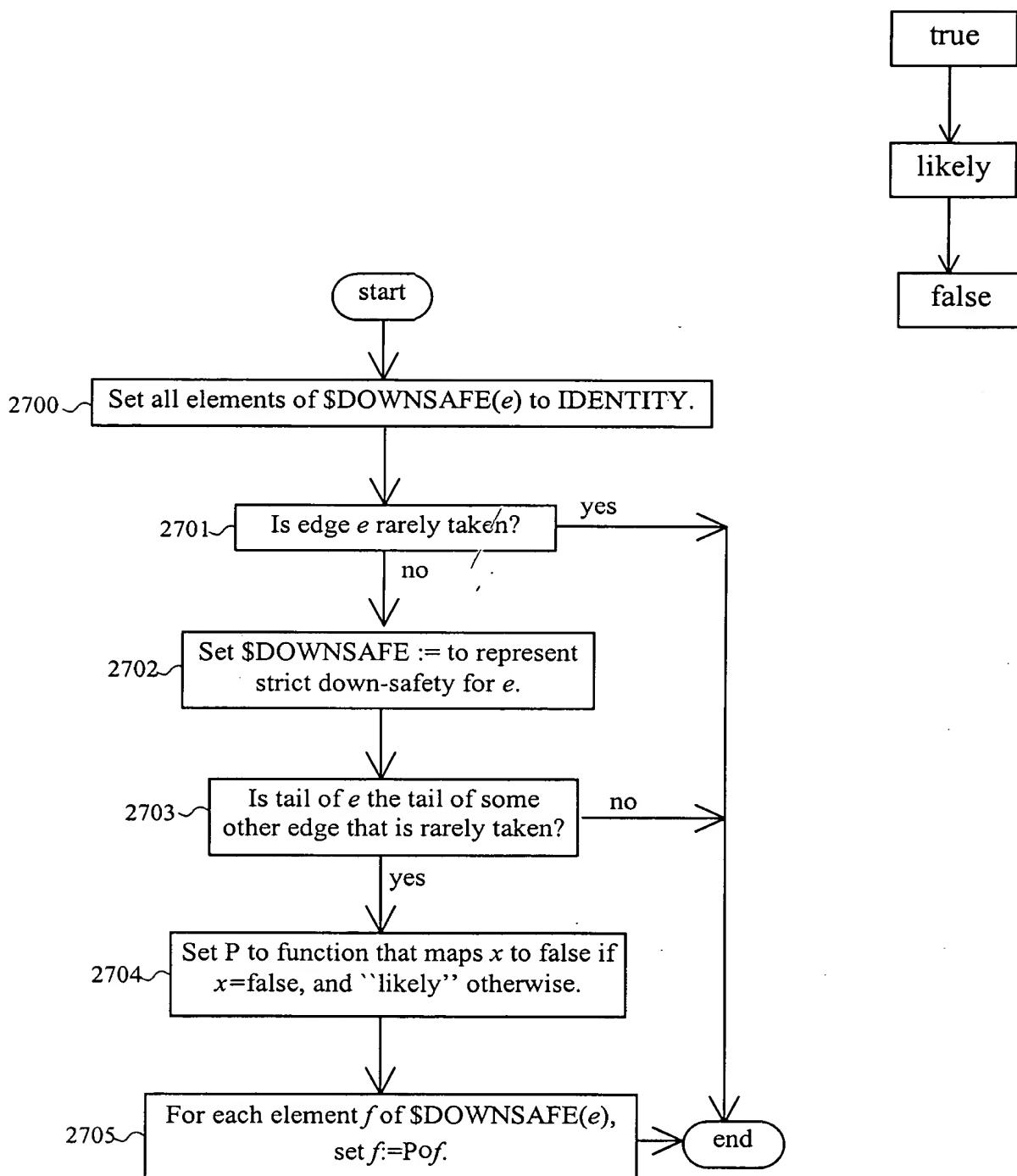
**FIG. 25**

FIG. 26**FIG. 27**

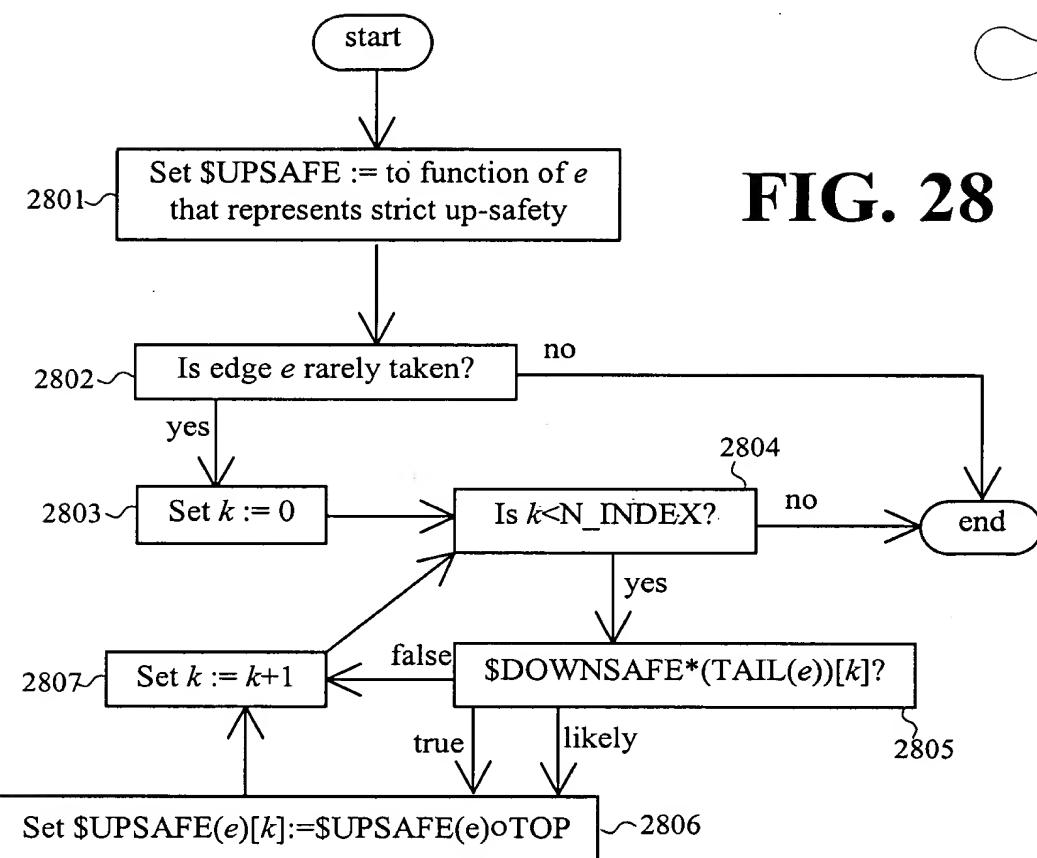
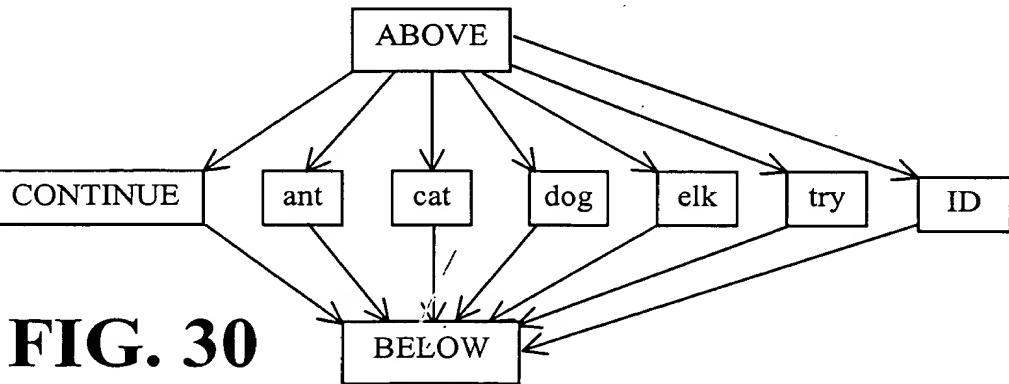
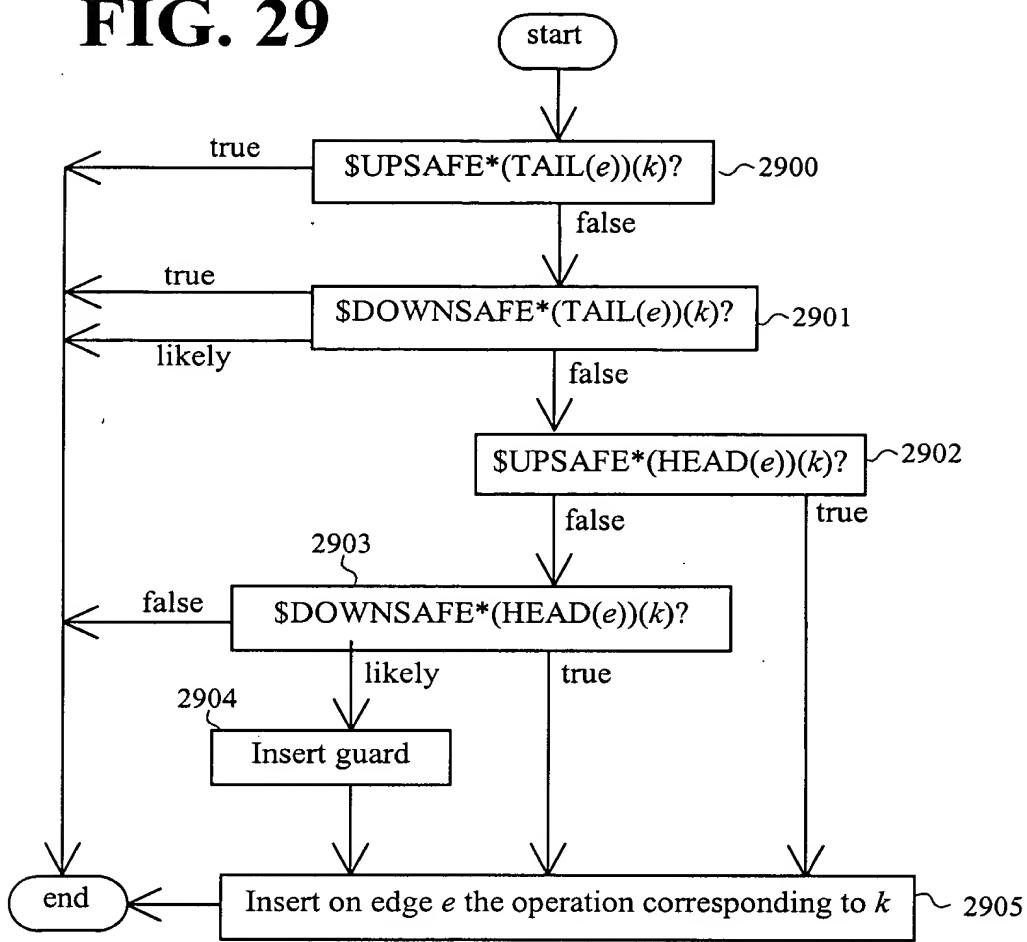
**FIG. 28**

FIG. 29**FIG. 30**